Utilização de redes neurais artificiais para a predição em bolsas de valores através de séries temporais

Acadêmico: Fernando Demarchi Natividade Luiz

Professor orientador: Valmei Abreu Junior

Professor coorientador: Miguel Diogenes Matrakas

Sumário

Introdução - contextualização

- Informatização do mercado acionário;
- Popularização da Internet;
- Acesso a dados históricos;
- Redes Neurais Artificiais (RNAs).

Introdução - justificativa

- Maior lucratividade possível para os investidores;
- Crescimento das RNAs aplicadas a importantes áreas do contexto social;
- Aperfeiçoamento de arquiteturas das RNAs;
- Crescimento de aplicações que utilizam técnicas de Inteligência Artificial.

Introdução - objetivos

Objetivo Geral: Especificar e aplicar um modelo de RNA para realizar predições nos valores de abertura das ações na bolsa de valores NASDAQ.

Objetivos Específicos:

- Especificar uma arquitetura de RNA;
- Definir um ambiente de desenvolvimento;
- Utilizar uma Interface de programação e aplicações (API) para a coleta dos dados que serão analisados;
- Treinar o modelo de rede especificado;
- Realizar testes com o modelo treinado, através das ações utilizadas;
- Analisar os resultados obtidos pela implementação, comparando-os com os seus resultados reais e avaliando sua capacidade de precisão.

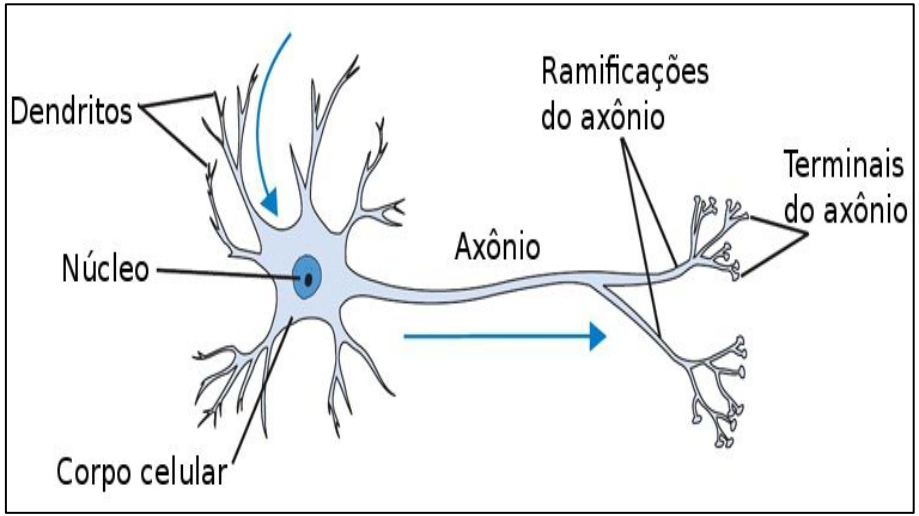
Revisão bibliográfica - Referências

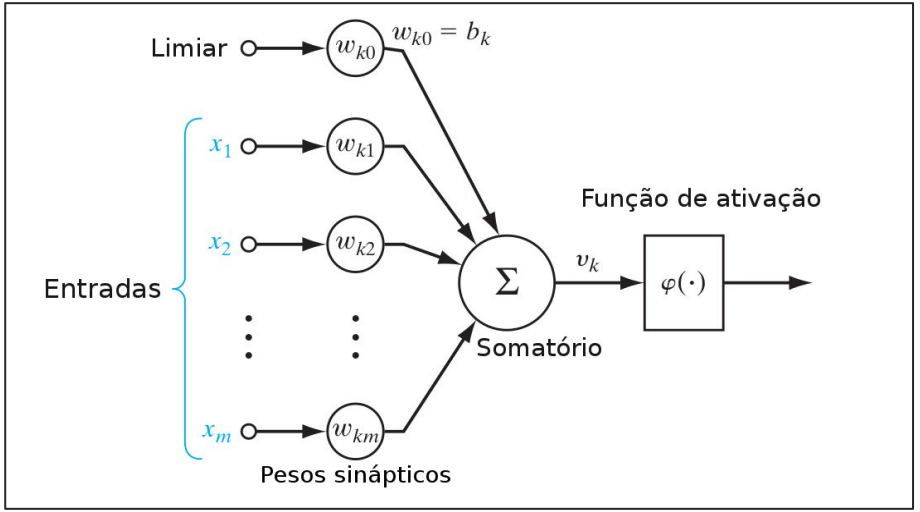
- WHITE, H. Economic prediction using neural networks: The case of IBM daily stock returns. IEEE International on Neural Networks. 2, p.451–458, 1988.

- KAMIJO, K.; TANIGAWA T. Stock Price Pattern Recognition A Recurrent Neural Network Approach. C C Information Technology Research Laboratories, NEC Corporation, 1990.

- DING, X. et al. Neural network based models for forecasting, em Proceedings of ADT'95. Wiley and Sons, p.243–252, 1995.

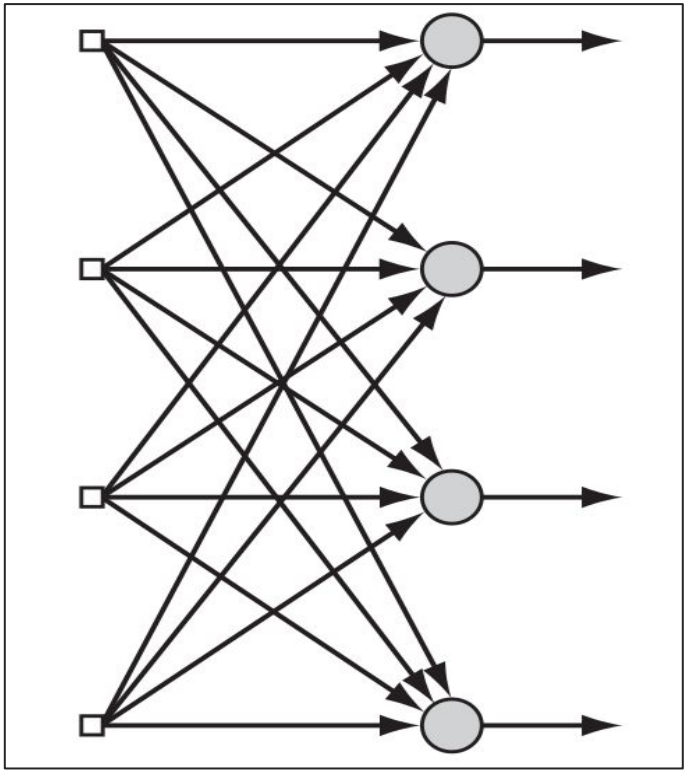
- LU, W. et al. The modeling of time series based on fuzzy information granules, Expert Systems with Applications. p.3799-3808, 2014.


Dendritos
Ramificações
do axônio
Terminais
do axônio
Axônio
Núcleo
Corpo celular


Limiar
w_{k0}
$w_{k0} = b_k$
x_1
w_{k1}
x_2
w_{k2}
Entradas
x_m
w_{km}
Pesos sinápticos
Σ
Somatório
v_k
Função de ativação
$\varphi(\cdot)$

Fundamentação teórica - Redes Neurais Artificiais

Rede Neural Artificial: Ferramenta computacional que visa trabalhar com os mesmos conceitos de um sistema nervoso, simulando o comportamento de um conjunto de neurônios biológicos através de neurônios artificiais (HAYKIN, 2009).



Características de arquitetura: A arquitetura de uma RNA pode ser definida pela quantidade de camada ocultas e pelo sentido do fluxo de dados.

Distribuição de camadas:

- Camada única (*single-layer*);
- Multicamadas (*multilayer*).

Fluxo de dados:

- Alimentada adiante (*feedforward*);
- Recorrente (*feedback*).

O processamento em cada neurônio se dá através da função de ativação.

- As funções de ativação podem ser classificadas como lineares e não-lineares

Segundo Haykin (2000), as funções que se destacam são:

- Função limiar: $y_k = 1 \ se \ x_j > 0$ $y_k = 0 \ se \ x_j < 0$
- Função sigmóide:
  - Função crescente;
  - Adequar componentes lineares e não-lineares em um intervalo [0,1] ; $f(x) = \frac{1}{1 + e^{-\lambda x}},$
- Tangente Hiperbólica.

Consiste em minimizar uma função de custo, por um algoritmo, através de iterações.

Componentes necessários para realizar o treinamento de uma RNA:

- Processo de Aprendizagem;
- Função de custo;
- Algoritmo de aprendizagem.

Características de um processo de aprendizagem:

- Inúmeras iterações;
  - Um limite de épocas estipulado;
  - Função de custo: Erro quadrático médio, raiz do erro quadrático médio ou o erro absoluto médio.
- Mapeamento dos dados;
- Atualização dos pesos sinápticos de cada neurônio;
- Aprendizagem supervisionada;
- Aprendizagem não supervisionada.

Características de algoritmos de aprendizagem

- Minimização local;
  - Convergem para mínimos locais.
- Minimização global;
  - Utilizam métodos de busca, como algoritmos genéticos.
- Algoritmo de retropropagação (*backpropagation*);
- Algoritmo de Aprendizado Hebbiano.

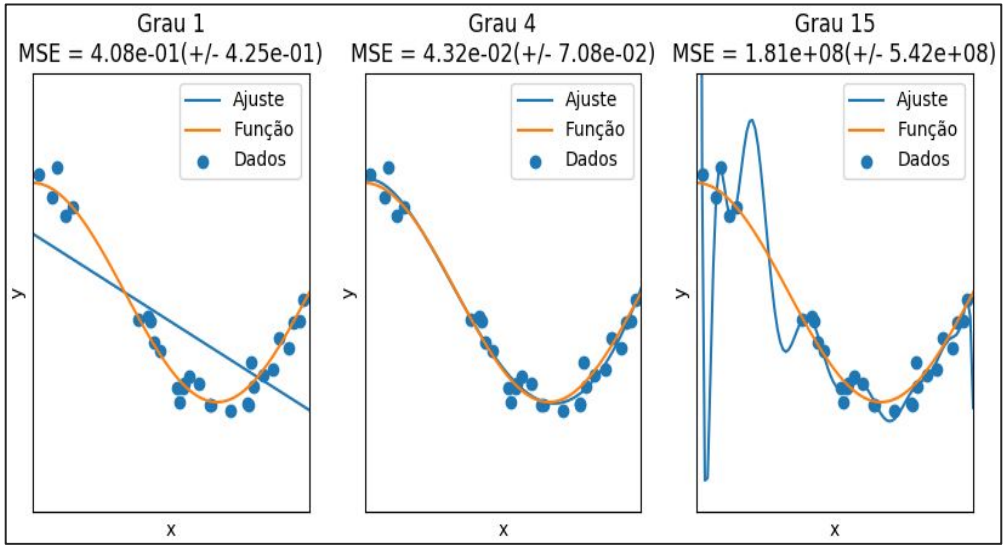
Características do *backpropagation:*

- Aprendizado supervisionado;
- Alimentação à frente;
- Função de ativação do tipo sigmóide;
- Coeficiente de aprendizado (*learning rate*);
- Propagação do erro para as camadas intermediárias;
- Atualização dos pesos sinápticos dos neurônios.

Fundamentação teórica - Treinamento de uma RNA: *Generalização*

Processo de adaptabilidade do modelo de RNA para padrões desconhecidos.

- Sobre-ajuste (*overfitting*): Alta taxa de variância;
- Sub-ajuste (*underfitting*): Não consegue mapear o modelo de dados;
- Sobretreinamento (*overtrainning*);
- Teoria da regularização.



Fundamentação teórica - Treinamento de uma RNA: *Séries Temporais*

Séries temporais são definidas como sequências de dados indexados ao longo de um período (MENDONCA NETO, 2014).

- A ordem temporal dos dados tem relevância na obtenção dos resultados.

Sistema de distribuição constituído por instituições financeiras que visa proporcionar um aumento no processo de capitalização das empresas. (TORORADAR, 2015).

- Análise fundamentalista de ações:
  - Leva em consideração a saúde financeira da empresa.

- Análise técnica de ações;
  - Utiliza como base o passado para gerar as perspectivas futuras.

- Bolsa de valores NASDAQ.
  - Segunda maior bolsa de valores do mundo;
  - Opera 100% de forma eletrônica;
  - Considerada como a “Nova Economia”.

Materiais e métodos - Seleção das ações

- Apple (AAPL);
- Amazon (AMZN);
- Intel (INTC);
- Microsoft (MSFT).

Materiais e métodos - Definição das séries

- Abertura: Valor da ação no ínicio do dia;
- Máximo: Valor máximo negociado no dia;
- Mínimo: Valor mínimo negociado no dia;
- Volume: Número de negociações do papel no dia (valor bruto das negociações);
- Fechamento: Valor de fechamento no dia;
- Médias móveis: indicador de tendência sequencial (10 e 26 dias);
- *Moving Average Convergence Divergence* (MACD): Indicador que controla a divergência e convergência das médias móveis.

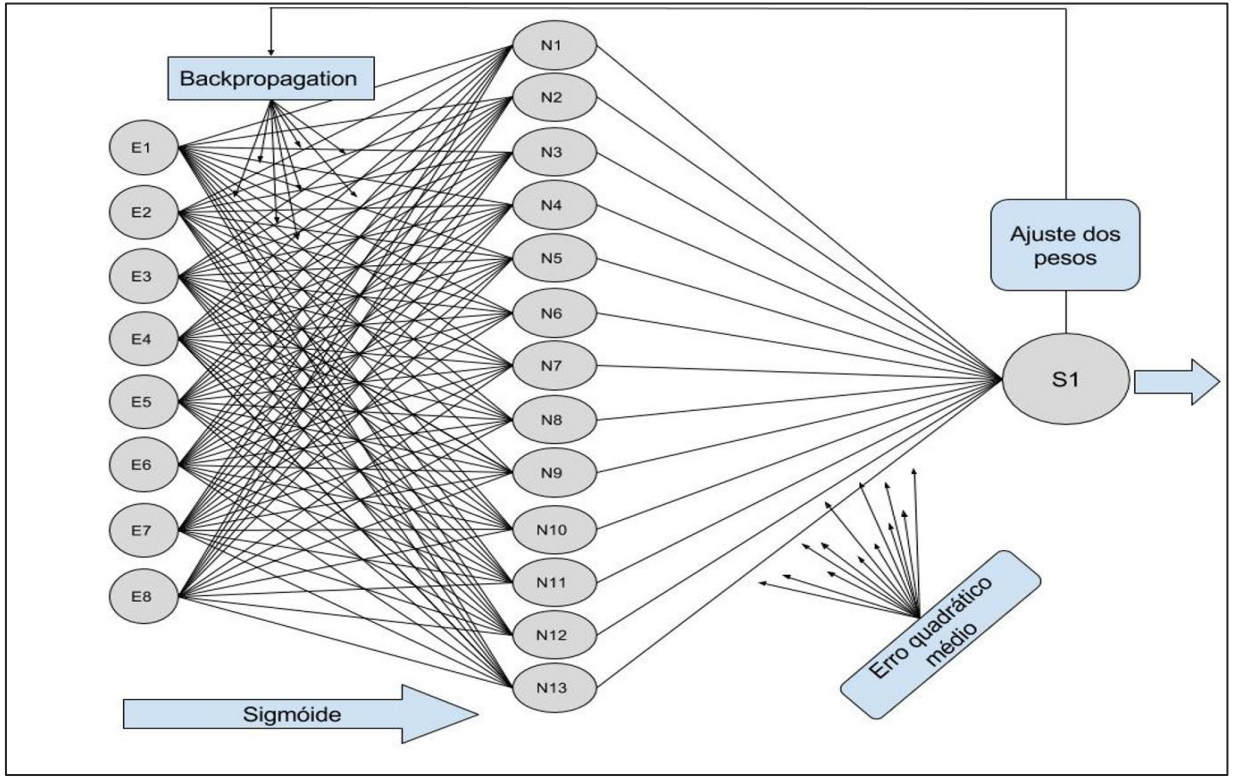
Materiais e métodos - Ferramentas para a coleta dos dados

- Linguagem de programação Python;
- Biblioteca pandas;
- Google *Finance* API;
- PyCharm IDE;
- matplotlib.

Materiais e métodos - Especificação do modelo de RNA: Definição da arquitetura

- Fluxo dos dados: Alimentada adiante;
- Aprendizado Supervisionado;
- *Perceptron* de Múltiplas Camadas (MLP, *Multilayer Perceptron*);
- Algoritmo *Backpropagation* para treinamento e aprendizado.

Materiais e métodos - Especificação do modelo de RNA: Definição da arquitetura

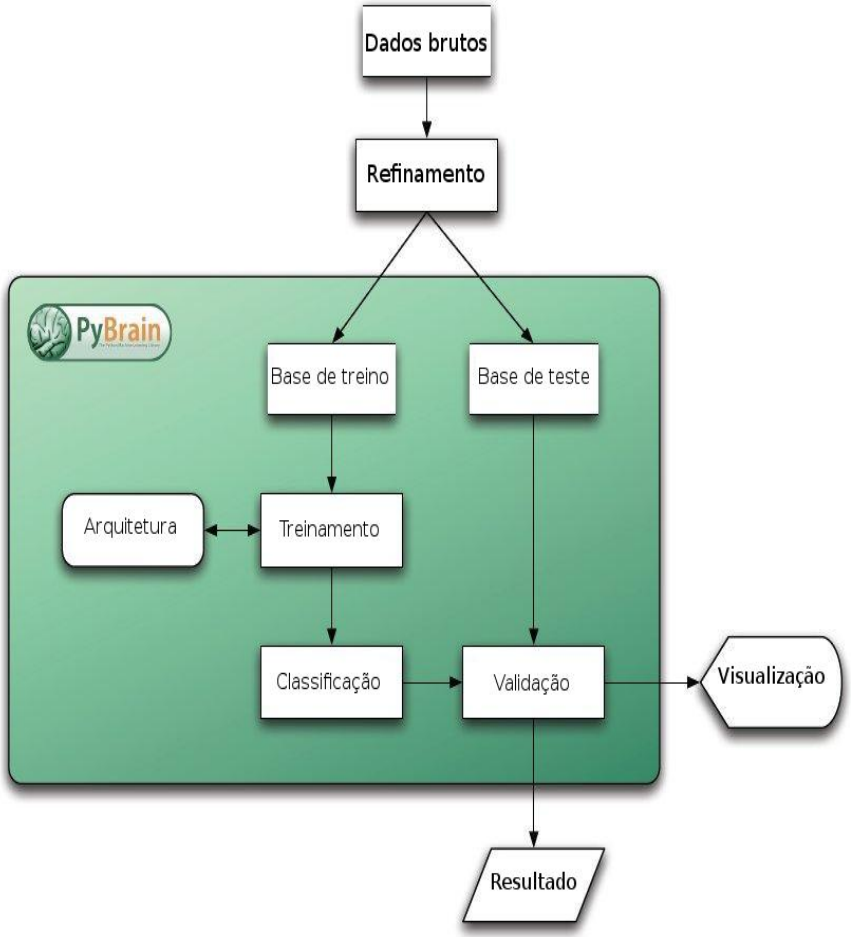


Materiais e métodos - Biblioteca PyBrain

É uma biblioteca de aprendizagem de máquinas para Python.

Características da biblioteca:

- Fornecer algoritmos flexíveis e fáceis de usar;
- Nível básico ao avançado;
- Possui algoritmos para: redes neurais, aprendizagem supervisionada e não supervisionada;
- Código aberto.



Materiais e métodos - Configuração do Hardware

- Intel(R) Core(TM) i5-4200U;
- 2.60 *Gigahertz;*
- *Memória de acesso aleatório (RAM) DDR3 com 6* gigabytes.

Implementação das técnicas - Realização da coleta de dados

- Desenvolvimento das classes para a coleta dos dados;

- Desenvolvimento de um Crawler responsável pela coleta das ações (09/04/2001 - 31/08/2017).

Implementação das técnicas - Realização da coleta de dados

```python
import abc
from seriestemporais.stack_empresas.crawler import *

class Empresa(metaclass=abc.ABCMeta):

    @abc.abstractmethod
    def executa_busca(self):
        pass

class Apple(Empresa):

    def __init__(self):
        self.nome_empresa = 'apple'
        self.codigo = 'AAPL'
        self.executa_busca()

    def executa_busca(self):
        crawler = Crawler(self.nome_empresa, self.codigo)
        crawler.executa_busca()
```

Implementação das técnicas - Realização da coleta de dados

```
"""
Crawler.py: Buscador genérico das séries temporais.
"""
__author__ = 'Fernando Demarchi Natividade Luiz'
__email__ = "nativanando@gmail.com"
__version__ = "0.0.1"

import pandas_datareader.data as web
import datetime

class Crawler:

    def __init__(self, nome_empresa, codigo):
        self.start = datetime.datetime(2001, 1, 1)
        self.end = datetime.datetime(2017, 8, 31)
        self.nome_empresa = nome_empresa
        self.codigo = codigo

    def executa_busca(self):
        file = web.DataReader(self.codigo, 'google', self.start, self.end)
        file.to_csv('~/Documentos/TCC/dist-tcc/Implementacao/dados/' + self.nome_empresa + '.csv')
```

Implementação das técnicas - Explorando o *Dataframe*

Cálculo das médias móveis de 10 e 26 dias: $$M_t = \frac{Z_t + Z_{t-1} + \cdots + Z_{t-k+1}}{k},$$

```
1 def executa_calculo_media_movel(self, indice_dataset, dia):
2     indice_dataset = indice_dataset + 1
3     indice_inicial = indice_dataset - (dia)
4     valor_media = 0
5     for i in range(indice_dataset):
6         if i >= indice_inicial:
7         valor_media = valor_media + self.dataset.loc[i].Close
8     self.dataset.set_value(indice_dataset - 1, 'movel_' + str(dia), ↩
    valor_media / dia, takeable=False)
9     self.dataset.to_csv('~/Documentos/TCC/dist-tcc/Implementacao/dados/'+↩
    self.nome_empresa+'_calculado.txt')
```

Implementação das técnicas - Explorando o *Dataframe*

Cálculo do MACD:

```
1 def calcula_macd(self, dataset):
2     dataset['MACD'] = dataset['movel_10'].sub(self.dataset['movel_26'])
3     return dataset
```

Implementação das técnicas - Explorando o *Dataframe*

Dataframe após o cálculo dos indicadores técnicos:

```
id,Date,Open,High,Low,Close,Volume,movel_26,movel_10,MACD
0,2001-04-09,11.84,11.9,10.7,11.18,23281900,10.551923076900001,9.747,-0.8049230769000015
1,2001-04-10,11.24,13.5,11.2,12.01,18532400,10.5284615385,9.784,-0.7444615385000013
2,2001-04-11,13.1,13.75,12.18,13.32,13214100,10.5838461538,10.036,-0.5478461538000001
3,2001-04-12,13.05,15.01,13.0,14.67,11266000,10.676923076900001,10.503,-0.1739230769000013
4,2001-04-16,14.5,14.54,13.65,14.03,6126500,10.766923076900001,10.883,0.11607692309999786
5,2001-04-17,13.81,15.63,13.75,14.74,4840200,10.8626923077,11.447,0.5843076923000012
6,2001-04-18,15.66,18.16,15.55,16.54,19287400,11.0903846154,12.238,1.1476153846
7,2001-04-19,16.9,16.9,15.69,15.99,8337400,11.2823076923,12.997,1.7146923077
8,2001-04-20,15.62,16.2,14.97,15.78,8401000,11.4807692308,13.663,2.1822307692000003
9,2001-04-23,16.39,17.46,16.02,16.2,11567900,11.6853846154,14.446,2.7606153845999994
10,2001-04-24,16.44,17.62,15.48,15.68,6168400,11.8653846154,14.896,3.0306153845999986
11,2001-04-25,15.69,16.1,14.6,16.09,9234300,12.0803846154,15.304,3.2236153846000004
12,2001-04-26,16.3,16.5,15.0,15.43,5884000,12.2796153846,15.515,3.235384615400014
13,2001-04-27,15.72,15.73,15.02,15.27,4497700,12.482307692300001,15.575,3.0926923076999984
14,2001-04-30,15.63,16.9,15.59,15.78,5479200,12.6973076923,15.75,3.0526923076999988
15,2001-05-01,15.9,17.05,15.8,16.89,4930300,12.955,15.965,3.01
16,2001-05-02,17.14,17.43,16.84,17.11,7437700,13.1923076923,16.022000000000002,2.8296923077000
17,2001-05-03,16.83,17.0,16.25,16.75,4962700,13.388846153800001,16.098,2.7091538461999978
18,2001-05-04,16.35,17.6,15.93,17.56,6122600,13.648846153800001,16.276,2.6271538461999993
19,2001-05-07,17.36,17.53,16.51,16.92,4983900,13.915,16.348,2.4330000000000003
20,2001-05-08,16.28,16.49,15.6,16.18,6290000,14.143846153800002,16.398,2.2541538461999977
21,2001-05-09,15.56,15.66,15.0,15.01,4604900,14.371153846199999,16.29,1.9188461538000008
22,2001-05-10,15.4,15.64,12.85,14.62,4832800,14.6015384615,16.209,1.607461538499999
23,2001-05-11,14.55,14.8,14.0,14.68,3341000,14.8430769231,16.15,1.3069230768999986
24,2001-05-14,14.53,14.53,13.15,13.33,6040300,15.005,15.905,0.8999999999999986
25,2001-05-15,13.35,14.3,13.14,13.54,6032800,15.2038461538,15.57,0.3661538461999996
26,2001-05-16,13.45,14.38,13.1,14.13,5678400,15.3173076923,15.272,-0.045307692299999765
27,2001-05-17,14.09,15.0,14.03,14.78,6220900,15.423846153800001,15.075,-0.3488461538000021
```

Implementação das técnicas - Normalização dos dados

Função de normalização:

$$L_n = (L_o - L_{min})/(L_{max} - L_{min})$$

```
1 def normaliza_coluna(self, dataset, coluna):
2     resultado = []
3     for i in range(dataset.__len__()):
4         resultado.append((dataset.iloc[i][coluna] - min(dataset[coluna]))↩
      / (max(dataset[coluna]) - min(dataset[coluna])))
5     dataset[coluna + '-normalizado'] = resultado
6     return dataset
```

Implementação das técnicas - Normalização dos dados

Função de desnormalização:

$$L_o = L_n * L_{max} + (1 - L_n) * L_{min}$$

```
1 def desnormaliza_valor(self, dataset, coluna, valor):
2     valor = valor * max(dataset[coluna]) + (1 - valor) * min(dataset[↩
    coluna])
3     return valor
```

Implementação das técnicas - Desenvolvimento da RNA

- *Script* para Criação de um Modelo *FeedForward;*
  - *pybrain.structure.networks.*
- *Organização das camadas do modelo;*
  - *pybrain.structure;*
    - LinearLayer, SigmoideLayer;
    - *addInputModule*;
    - *addModule;*
    - *AddOutPutModule*;
    - *FullConnection;*
    - sortModules.

Implementação das técnicas - Desenvolvimento da RNA

```python
from pybrain.structure import FeedForwardNetwork

class FeedForwardNetworkPyBrain:

    def __init__(self):
        self.rede = FeedForwardNetwork()
```

```python
from pybrain.structure import LinearLayer, SigmoidLayer

class FeedForwardNetworkPyBrainLayers:

    def __init__(self, tamanho_camada_entrada, tamanho_camada_oculta, tamanho_camada_saida):
        self.camada_entrada = LinearLayer(self.camada_entrada, name="entrada")
        self.camada_oculta = SigmoidLayer(self.camada_oculta, name="oculta")
        self.camada_saida = LinearLayer(self.camada_saida, name="saida")
```

Implementação das técnicas - Desenvolvimento da RNA

```
1 def adicionaEstrutura(self, rede):
2     rede.addInputModule(self.camada_entrada)
3     rede.addModule(self.camada_oculta)
4     rede.addOutputModule(self.camada_saida)
5     return rede
```

```
1 def adicionaConexoes(self):
2     from pybrain.structure import FullConnection
3     '''Importação do objeto FullConnection através do módulo pybrain.↩
    structure '''
4     self.ligacao_entrada_oculta = FullConnection(self.camada_entrada, ↩
    self.camada_oculta)
5     self.ligacao_oculta_saida = FullConnection(self.camada_oculta, self.↩
    camada_saida)
6     self.network.addConnection(self.ligacao_oculta_saida)
7     self.network.addConnection(self.ligacao_entrada_oculta)
```

Implementação das técnicas - Desenvolvimento da RNA

- *Treinamento da rede:*
  - *pybrain.datasets;*
  - *pybrain.supervised.*

```
def adicionaDadosTreinamento(self, nome_empresa):
    import pandas as pd
    from pybrain.datasets import SupervisedDataSet

    try:
        self.dataset = pd.read_csv('~/Documentos/TCC/dist-tcc/Implementacao/dados_calculados/'+ nome_empresa + '_normalizado.txt', header=0)
    except IOError:
        print ("Erro ao abrir os dados da empresa "+nome_empresa+"")
        return 0

    self.dataset_treino = SupervisedDataSet(8, 1)

    for i in range(self.dataset.__len__() - 8):
        self.dataset_treino.addSample([
        self.dataset.iloc[i]['Open-normalizado'],
        self.dataset.iloc[i]['High-normalizado'],
        self.dataset.iloc[i]['Low-normalizado'],
        self.dataset.iloc[i]['Close-normalizado'],
        self.dataset.iloc[i]['Volume-normalizado'],
        self.dataset.iloc[i]['movel_26-normalizado'],
        self.dataset.iloc[i]['movel_10-normalizado'],
        self.dataset.iloc[i]['MACD-normalizado']],
        self.dataset.iloc[i + 1]['Open-normalizado'])

    return self.dataset_treino
```

Implementação das técnicas - Desenvolvimento da RNA

- *Treinamento da rede:*
  - *BackPropTrainer;*
  - *pybrain.tools.customxml;*
  - *NetworkWriter.*

```
1 def realizaTreinamento(self, rede, dataset_treino, epocas, nome_empresa):
2     from pybrain.supervised import BackpropTrainer
3     from pybrain.tools.customxml import NetworkWriter
4
5     treinamento = BackpropTrainer(rede, dataset_treino, learningrate=0.4,↩
       verbose=True)
6     treinamento.trainEpochs(epochs = epocas)
7     NetworkWriter.writeToFile(rede, 'snapshot_redes/rede-feedforward-' + ↩
     nome_empresa + '.xml')
```

Análise dos resultados

Período de treinamento: 09/04/2001 até 21/08/2017;

Período de predição: 23/08/2017 a 31/08/2017;

Cenários analisados:

- Ciclos de treinamento: 200 e 1000 iterações.

Métricas analisadas:

- Validação do algoritmo de inicialização;
- Comportamento da função de custo (EQM);
- Erro relativo percentual do valor resultante;
- Erro médio e desvio padrão do período;
- Variação dos resultados em relação aos seus valores reais.

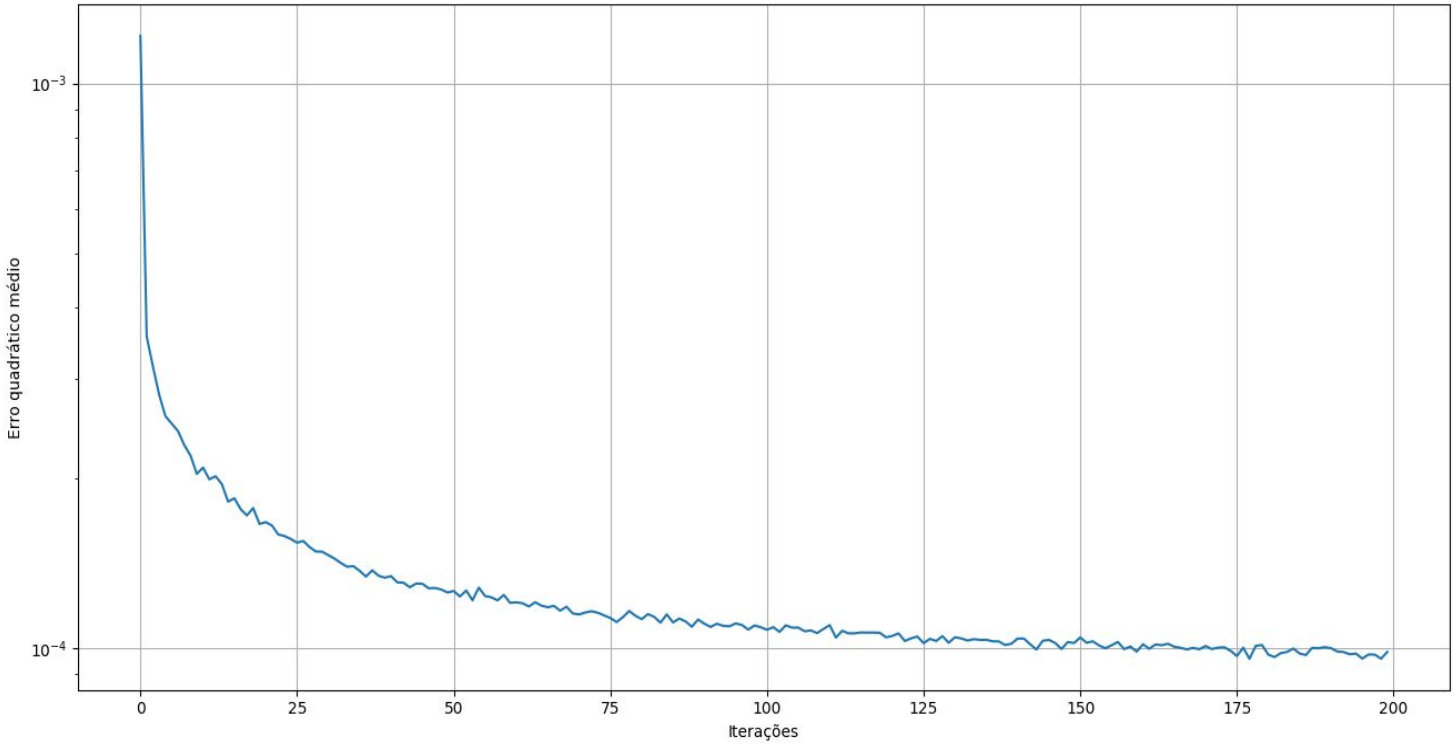
Análise dos resultados: Intel Corporation: 200 iterações

- Número de registros da série: 4117;
- Ciclos de treinamento: 200;
- Total de exemplos abstraídos pela rede: 823.400.

Execução do treinamento: 3 vezes:

1. EQM: 0.00248996652176 e 9.44359025338.10−05;
2. EQM: 0.00102981131422 e 9.1790170623.10−05;
3. EQM: 0.00026991101482 e 9.49959768553.10−05.

Análise dos resultados: Intel Corporation: 200 iterações



Análise dos resultados: Intel Corporation: 200 iterações

Dados utilizados para realizar os testes:

Data	Abertura	Alta	Baixa	Fechamento	Volume
23/08/2017	34.54	34.81	34.38	34.66	196.481,34
24/08/2017	34.70	34.89	34.55	34.71	143.018,92
25/08/2017	34.82	34.93	34.58	34.67	147.268,29
28/08/2017	34.78	34.80	34.59	34.65	207.128,76
29/08/2017	34.51	34.75	34.46	34.73	158.436,68
30/08/2017	34.75	34.96	34.63	34.89	185.650,07
31/08/2017	34.94	35.18	34.87	35.07	163.667,72

Análise dos resultados: Intel Corporation: 200 iterações

Resultados obtidos pela RNA:

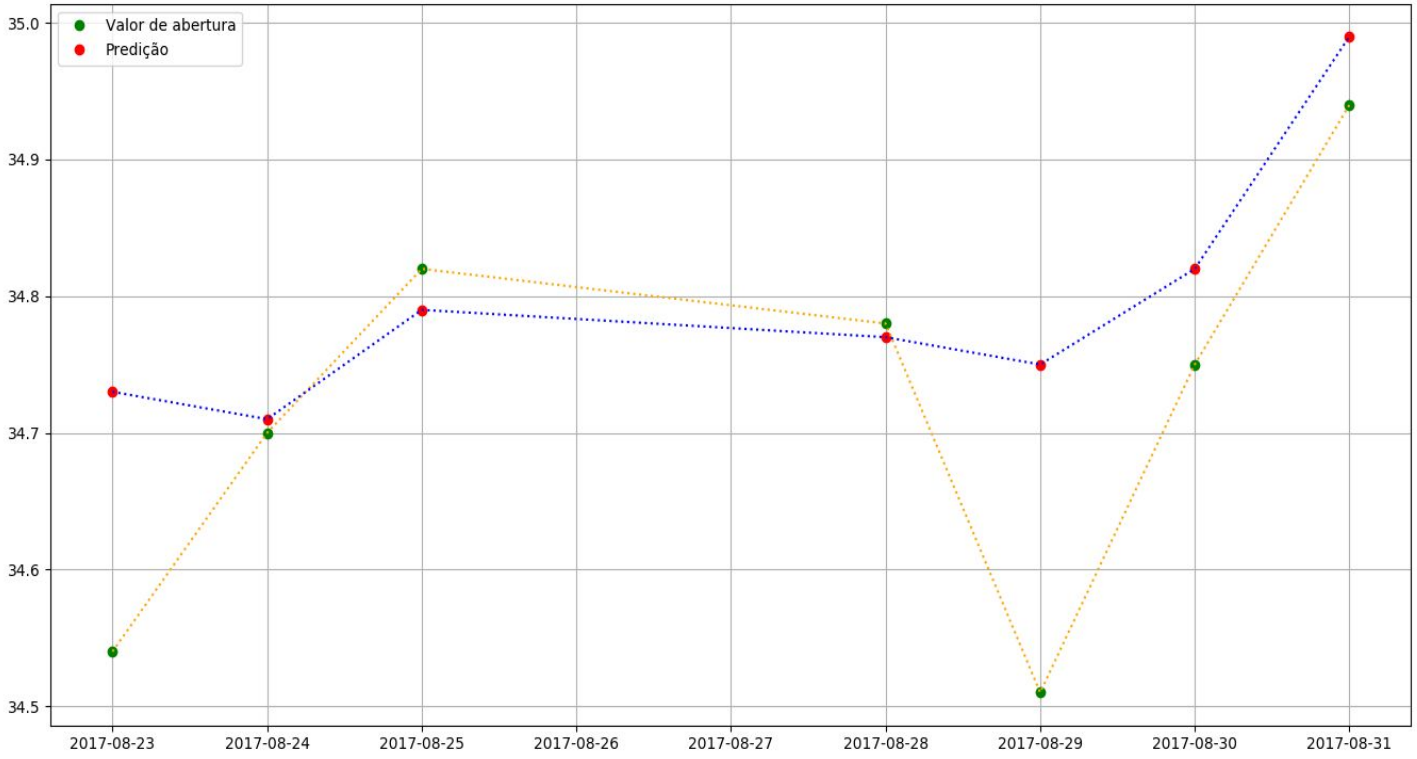
Data	Valor real	Resultado	Erro (%)	Variação
23/08/2017	34.54	34.73	0.550	-0.19
24/08/2017	34.70	34.71	0.028	-0.01
25/08/2017	34.82	34.79	0.086	0.03
28/08/2017	34.78	34.77	0.028	0.01
29/08/2017	34.51	34.75	0.695	-0.24
30/08/2017	34.75	34.82	0.201	-0.07
31/08/2017	34.94	34.99	0.143	0.05

Erro médio: 0.24%;

Desvio padrão: 0.26%;

Dispersão dos resultados em um momento de queda: 23/08 e 29/08.

Análise dos resultados: Intel Corporation: 200 iterações



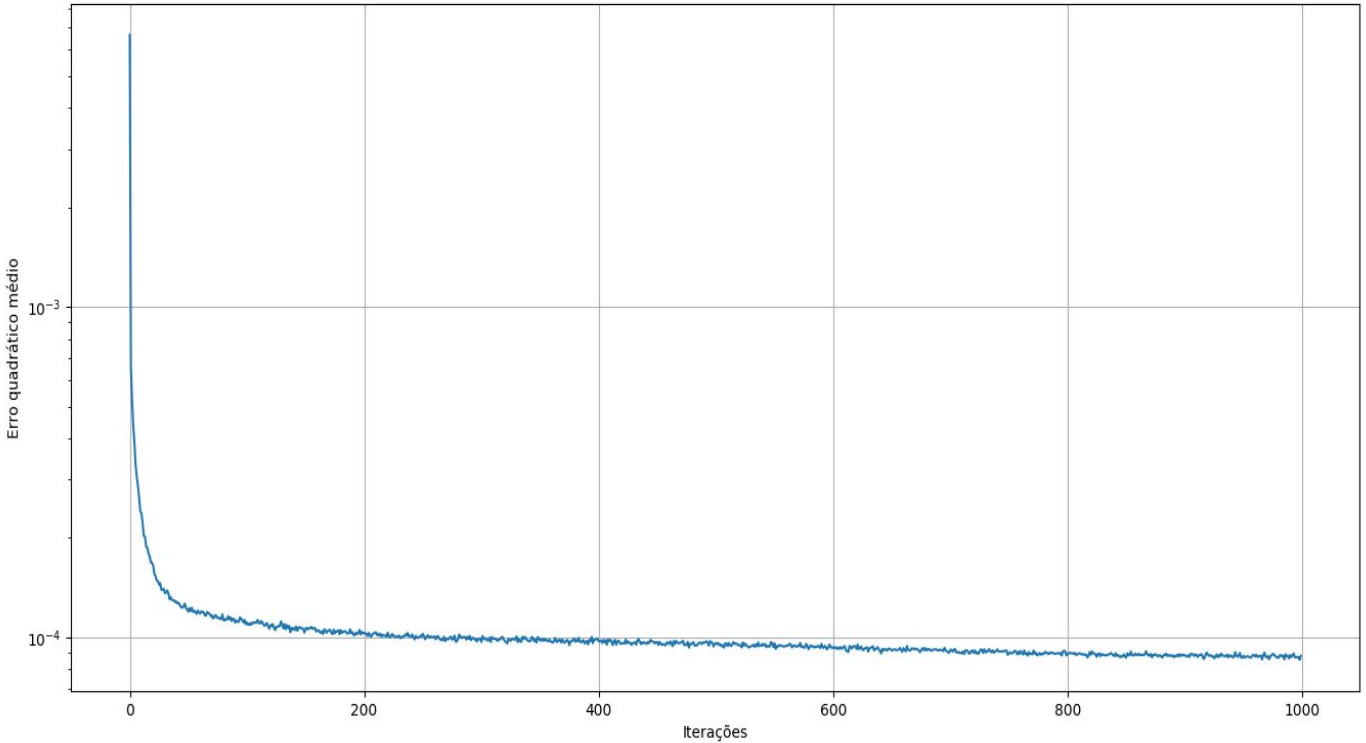
Análise dos resultados: Intel Corporation: 1000 iterações

- Número de registros da série: 4117;
- Ciclos de treinamento: 1000;
- Total de exemplos abstraídos pela rede: 4.117.000.

Execução do treinamento: 3 vezes:

1. EQM: 0.00664064049629 e 8.79435631218535 31.10−05;
2. EQM: 0.00260377914122 e 8.91402403614.10−05;
3. EQM: 0.00496238079711 e 8.90814371631.10−05.

Análise dos resultados: Intel Corporation: 1000 iterações



Análise dos resultados: Intel Corporation: 1000 iterações

Resultados obtidos pela RNA:

Data	Valor real	Resultado	Erro (%)	Variação
23/08/2017	34.54	34.67	0.376	-0.13
24/08/2017	34.70	34.64	0.172	0.06
25/08/2017	34.82	34.71	0.315	0.11
28/08/2017	34.78	34.68	0.287	0.10
29/08/2017	34.51	34.65	0.405	-0.14
30/08/2017	34.75	34.70	0.143	0.05
31/08/2017	34.94	34.84	0.286	0.10

Erro médio: 0.28%;

Desvio padrão: 0.09%;

Suavização dos erros, porém com menos exatidão. 23 e 29/08.

Análise dos resultados: Microsoft Corporation: 200 iterações

- Número de registros da série: 4117;
- Ciclos de treinamento: 200;
- Total de exemplos abstraídos pela rede: 823.400.

Execução do treinamento: 2 vezes:

1. EQM: 0.0018579672451418453 e 3.6623220378015219.10−05;

2. EQM: 0.0047598765727053707 e 3.8483854181673219.10−05.

Análise dos resultados: Microsoft Corporation: 200 iterações

Dados utilizados para realizar os testes:

Data	Abertura	Alta	Baixa	Fechamento	Volume
23/08/2017	72.96	73.15	72.53	72.72	137.665,07
24/08/2017	72.74	72.86	72.07	72.69	170.982,82
25/08/2017	72.86	73.35	72.48	72.82	127.943,01
28/08/2017	73.06	73.09	72.55	72.83	145.697,15
29/08/2017	72.25	73.16	72.05	73.05	114.783,82
30/08/2017	73.01	74.21	72.83	74.01	168.978,01
31/08/2017	74.03	74.96	73.80	74.77	276.528,11

Análise dos resultados: Microsoft Corporation: 200 iterações

Resultados obtidos pela RNA:

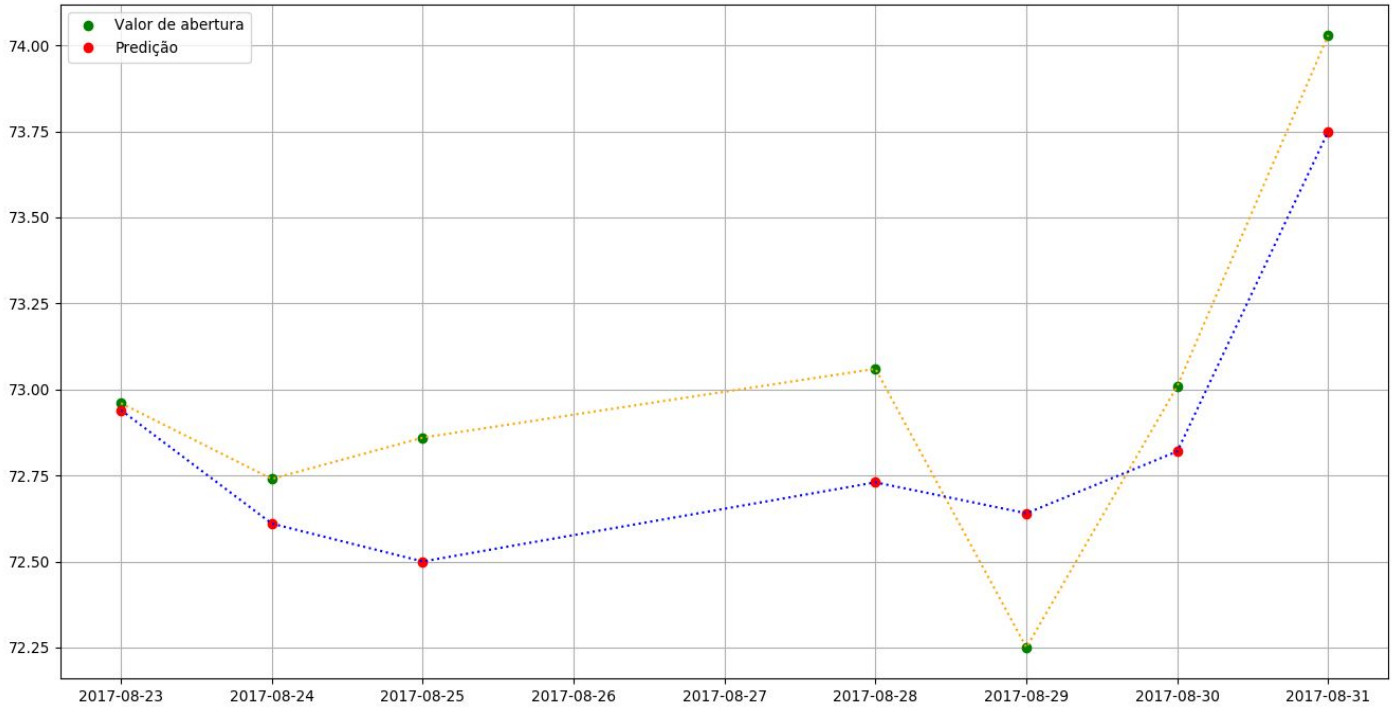
Data	Valor real	Resultado	Erro (%)	Variação
23/08/2017	72.96	72.94	0.027	0.02
24/08/2017	72.74	72.61	0.028	0.13
25/08/2017	72.86	72.50	0.086	0.36
28/08/2017	73.06	72.73	0.028	-0.33
29/08/2017	72.25	72.64	0.695	-0.39
30/08/2017	73.01	72.82	0.201	0.19
31/08/2017	74.03	73.75	0.143	0.28

Erro médio: 0.17%;

Desvio padrão: 0.23%;

Dispersão dos resultados em um momento de queda.

Análise dos resultados: Microsoft Corporation: 200 iterações



Análise dos resultados: Microsoft Corporation: 1000 iterações

- Número de registros da série: 4117;
- Ciclos de treinamento: 1000;
- Total de exemplos abstraídos pela rede: 4.117.000.

Execução do treinamento: 3 vezes:

1. EQM: 0.005757041736 e 3.0266887029028769.10−05;
2. EQM: 0.0035660777044 e 2.57225977588.10−05;
3. EQM: 0.00624617417834 e 2.5743685343480545.10−05.

Análise dos resultados: Microsoft Corporation: 1000 iterações

Resultados obtidos pela RNA:

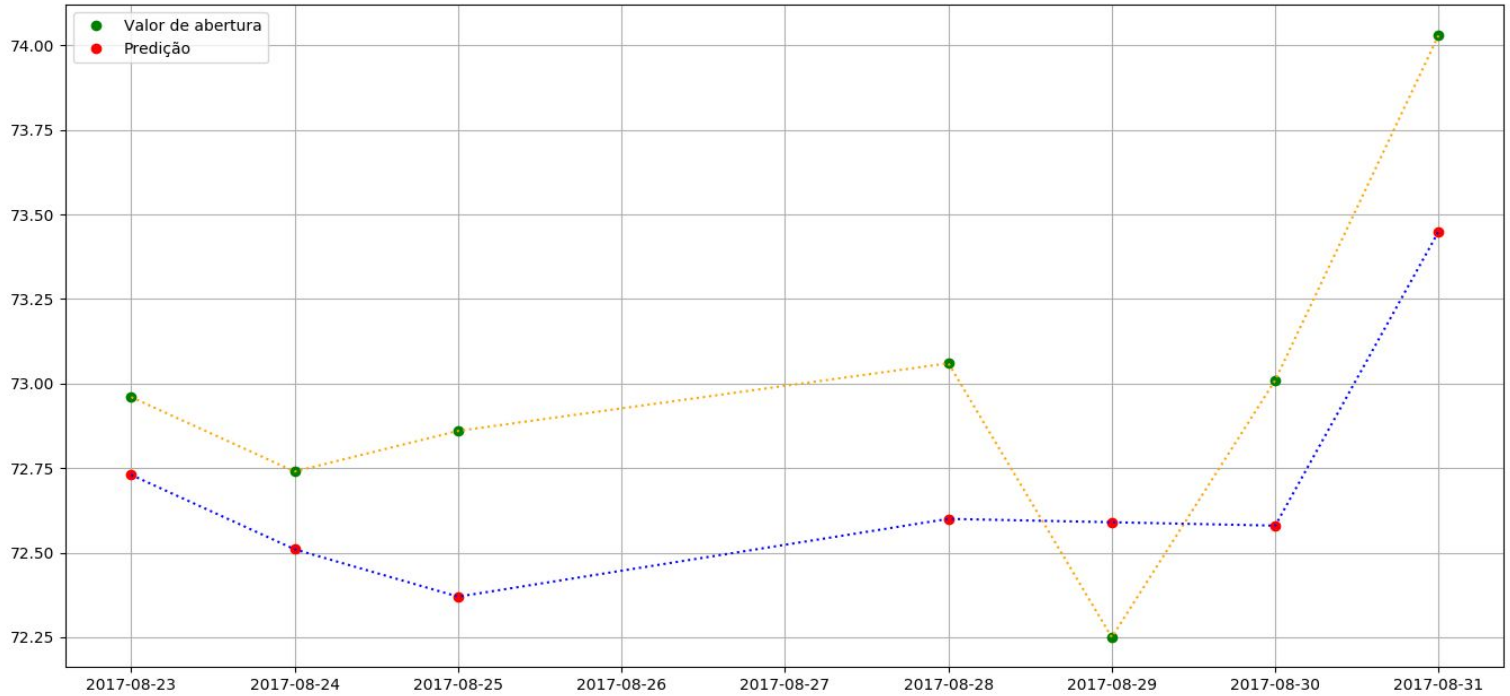
Data	Valor real	Resultado	Erro (%)	Variação
23/08/2017	72.96	72.73	0.315	0.23
24/08/2017	72.74	72.51	0.316	0.23
25/08/2017	72.86	72.37	0.672	0.49
28/08/2017	73.06	72.60	0.629	0.46
29/08/2017	72.25	72.59	0.470	-0.34
30/08/2017	73.01	72.58	0.588	0.43
31/08/2017	74.03	73.45	0.783	0.58

Erro médio: 0.53%;

Desvio padrão: 0.17%;

Dispersão dos resultados por uma quantidade excessiva de treinamento.

Análise dos resultados: Microsoft Corporation: 1000 iterações



Análise dos resultados: Apple Inc: 200 iterações

- Número de registros da série: 4117;
- Ciclos de treinamento: 200;
- Total de exemplos abstraídos pela rede: 823.400.

Execução do treinamento: 2 vezes:

1. EQM: 0.000439745663518 e 1.9971768385490857.10−05;

2. EQM: 0.000710657880333 e 1.915103371487409.10−05.

Análise dos resultados: Apple Inc: 200 iterações

Dados utilizados para realizar os testes:

Data	Abertura	Alta	Baixa	Fechamento	Volume
23/08/2017	159.07	160.47	158.88	159.98	193.990,81
24/08/2017	160.43	160.74	158.55	159.27	198.189,18
25/08/2017	159.65	160.56	159.27	159.86	254.800,63
28/08/2017	160.14	162.00	159.93	161.47	259.659,72
29/08/2017	160.10	163.12	160.00	162.91	295.169,10
30/08/2017	163.80	163.89	162.61	163.35	272.695,84
31/08/2017	163.64	164.52	163.48	164.00	267.850,96

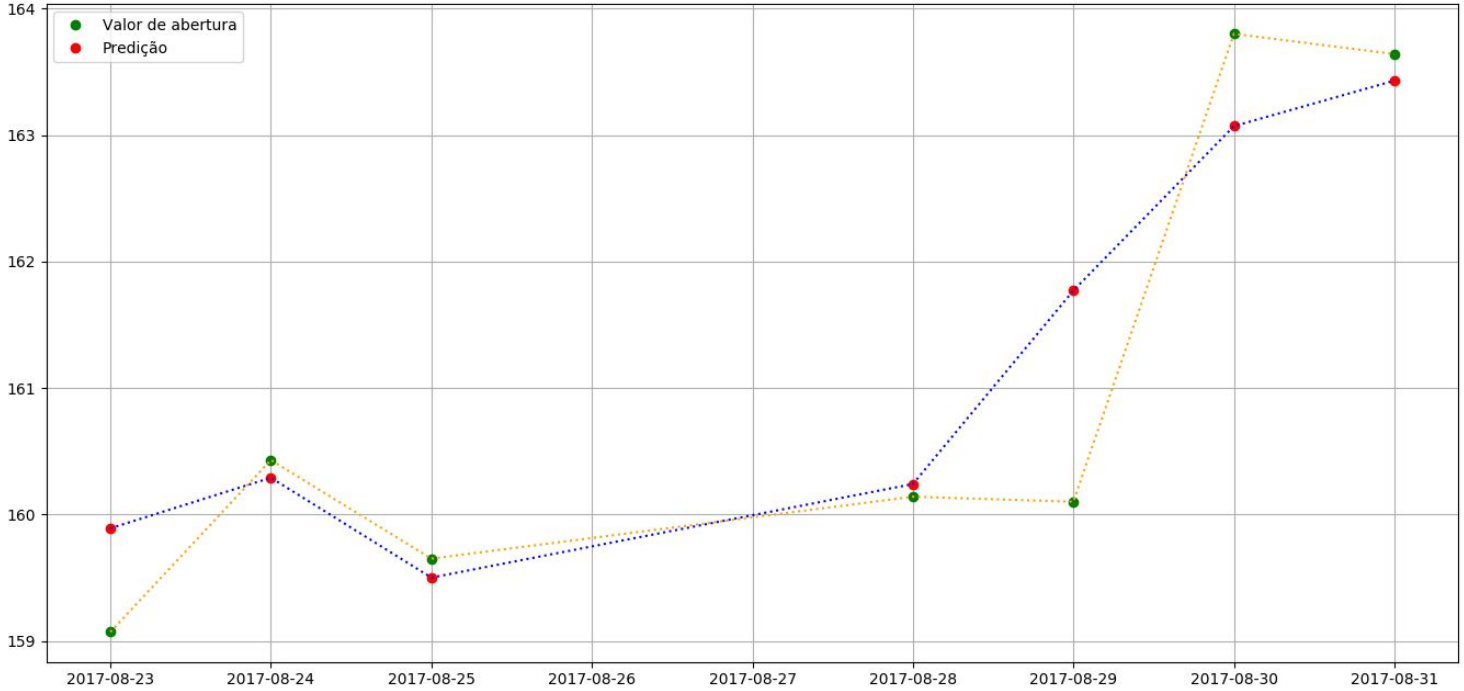
Análise dos resultados: Apple Inc: 200 iterações

Resultados obtidos pela RNA:

Data	Valor real	Resultado	Erro (%)	Variação
23/08/2017	159.07	159.89	0.515	0.82
24/08/2017	160.43	160.29	0.087	0.14
25/08/2017	159.65	159.50	0.093	0.15
28/08/2017	160.14	160.24	0.062	-0.10
29/08/2017	160.10	161.77	1.043	-1.67
30/08/2017	163.80	163.07	0.445	0.73
31/08/2017	163.64	163.43	0.128	0.21

Erro médio: 0.33%;
Desvio padrão: 0.36%;
Dispersão nos resultados: 23/08/2017 e 29/08/2017.

Análise dos resultados: Apple Inc: 200 iterações



Análise dos resultados: Apple Inc: 1000 iterações

- Número de registros da série: 4117;
- Ciclos de treinamento: 1000;
- Total de exemplos abstraídos pela rede: 4.117.000.

Execução do treinamento: 3 vezes:

1. EQM: 0.000868275992533 e 1.5728321897054983.10−05;
2. EQM: 0.00411352897345 e 1.60710693661.10−05;
3. EQM: 0.000326093399269 e 1.85386451723.10−05.

Análise dos resultados: Apple Inc: 1000 iterações

Resultados obtidos pela RNA:

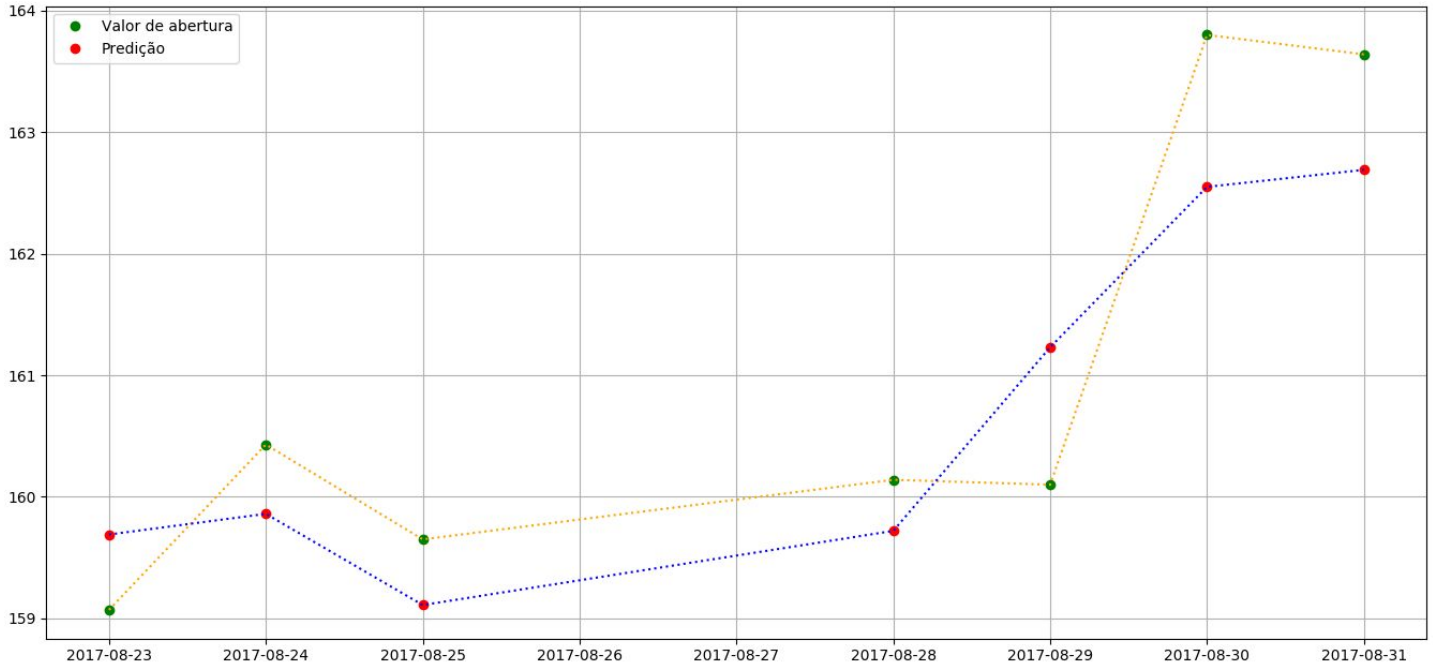
Data	Valor real	Resultado	Erro (%)	Variação
23/08/2017	159.07	159.69	0.389	-0.62
24/08/2017	160.43	159.86	0.355	0.57
25/08/2017	159.65	159.11	0.338	0.54
28/08/2017	160.14	159.72	0.262	0.42
29/08/2017	160.10	161.23	0.705	-1.13
30/08/2017	163.80	162.55	0.763	1.25
31/08/2017	163.64	162.69	0.580	0.95

Erro médio: 0.53%;

Desvio padrão: 0.19%;

Suavizou os erros mais altos: 23/08 e 29/08/2017

Análise dos resultados: Apple Inc: 1000 iterações



Análise dos resultados: Amazon Inc: 200 iterações

- Número de registros da série: 4117;
- Ciclos de treinamento: 200;
- Total de exemplos abstraídos pela rede: 823.400.

Execução do treinamento: 3 vezes:

1. EQM: 0.00240540733964 e 1.79339271141.10−05;
2. EQM: 0.00489046785604 e 1.59601609973.10−05;
3. EQM: 0.00849440933377 e 1.77038646762.10−05.

Análise dos resultados: Amazon Inc: 200 iterações

Dados utilizados para realizar os testes:

Data	Abertura	Alta	Baixa	Fechamento	Volume
23/08/2017	959.38	962.00	954.20	958.00	266.826,40
24/08/2017	957.42	959.00	941.14	952.45	519.572,60
25/08/2017	956.00	957.62	944.10	945.26	332.479,10
28/08/2017	946.54	953.00	942.25	946.02	259.673,70
29/08/2017	940.00	956.00	936.33	954.06	287.429,90
30/08/2017	958.44	969.41	956.91	967.59	290.460,40
31/08/2017	974.70	981.00	972.76	980.60	333.148,80

Análise dos resultados: Amazon Inc: 200 iterações

Resultados obtidos pela RNA:

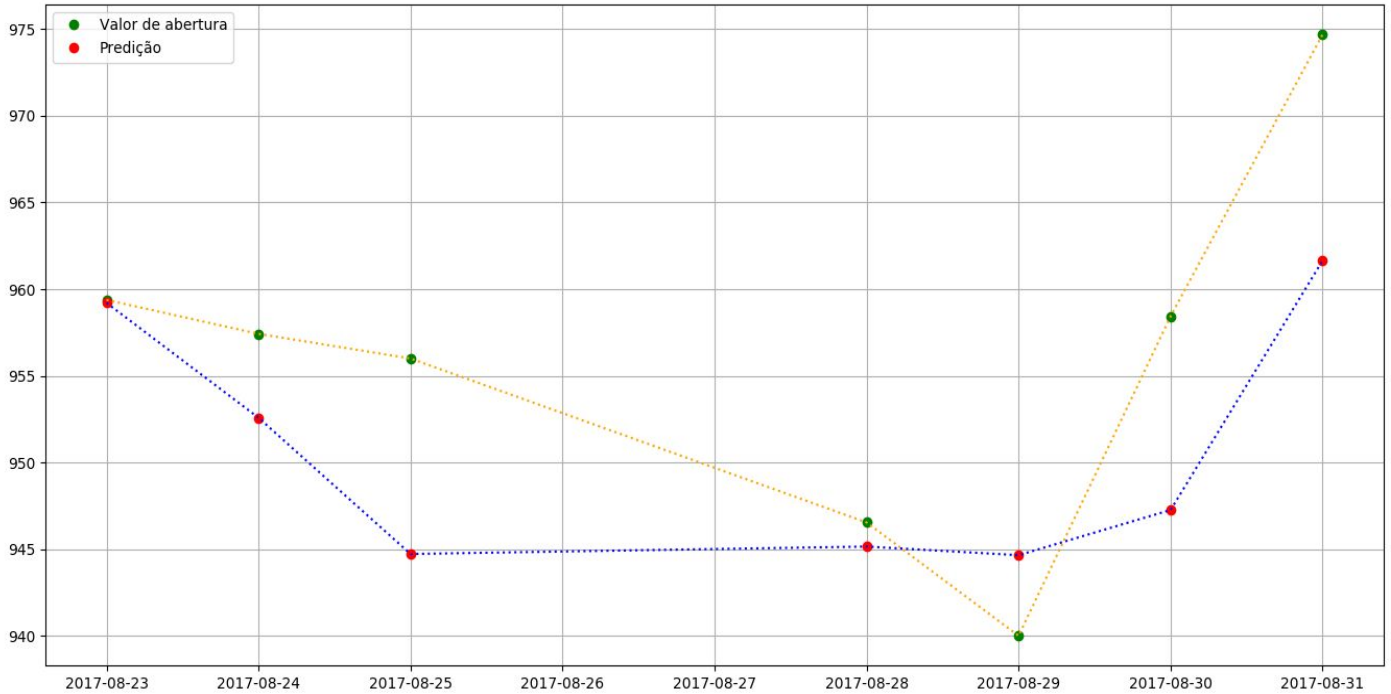
Data	Valor real	Resultado	Erro (%)	Variação
23/08/2017	959.38	959.21	0.017	0.17
24/08/2017	957.42	952.58	0.505	4.84
25/08/2017	956.00	944.72	1.179	11.28
28/08/2017	946.54	945.15	0.146	1.39
29/08/2017	940.00	944.65	0.494	-4.65
30/08/2017	958.44	947.26	1.166	11.18
31/08/2017	974.70	961.65	1.338	13.05

Erro médio: 0.69%;

Desvio padrão: 0.53%;

Dispersão nos resultados: 25/08/2017, 30/08/2017 e 31/08/2017.

Análise dos resultados: Amazon Inc: 200 iterações



Análise dos resultados: Amazon Inc: 1000 iterações

- Número de registros da série: 4117;
- Ciclos de treinamento: 1000;
- Total de exemplos abstraídos pela rede: 4.117.000.

Execução do treinamento: 3 vezes:

1. EQM: 0.0007172644682663635 e 1.2545666972429883.10−05;
2. EQM: 0.0012558318171033 49 e 1.607023747635374.10−05;
3. EQM: 0.0022901768887634102 e 1.363795522772317.10−05.

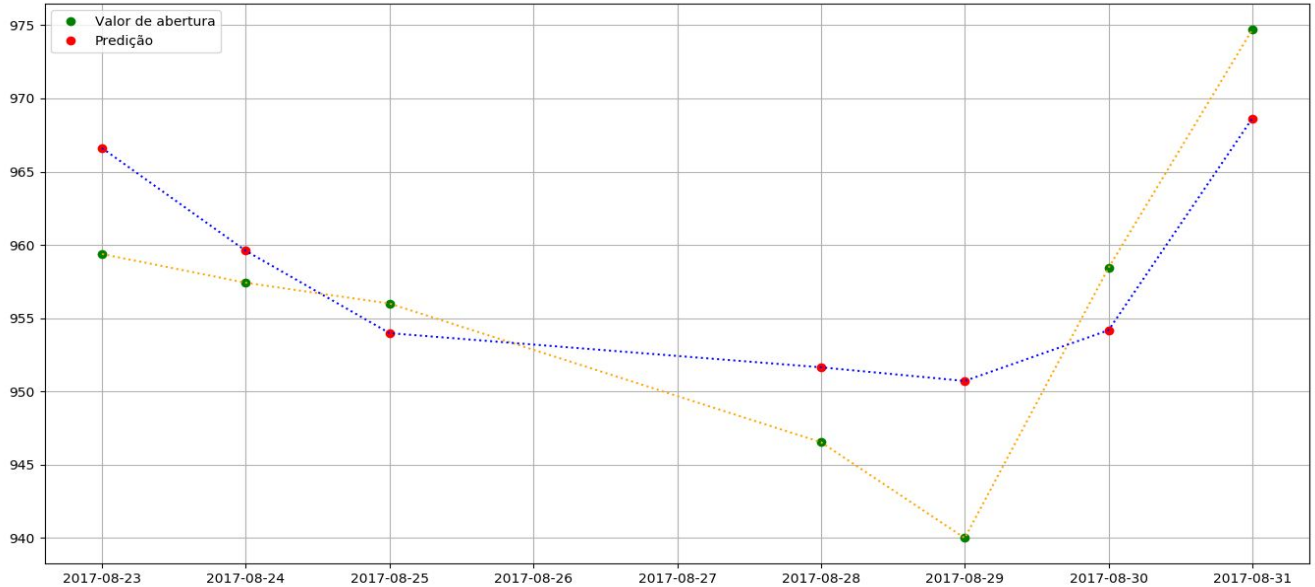
Análise dos resultados: Amazon Inc: 1000 iterações

Resultados obtidos pela RNA:

Data	Valor real	Resultado	Erro (%)	Variação
23/08/2017	959.38	966.61	0.753	-7.23
24/08/2017	957.42	959.60	0.227	-2.18
25/08/2017	956.00	953.98	0.211	2.02
28/08/2017	946.54	951.65	0.539	-5.11
29/08/2017	940.00	950.72	1.140	-10.72
30/08/2017	958.44	954.18	0.444	4.26
31/08/2017	974.70	968.64	0.621	6.06

Erro médio: 0.56%;
Desvio padrão: 0.32%;
Suavização dos erros altos obtidos no treinamento anterior: 25, 30 e 31/08;
Erro médio inferior ao de 200 iterações.

Análise dos resultados: Amazon Inc: 1000 iterações



Análise dos resultados: Tempo da execução

- 5 minutos para 200 Iterações;
- 28 minutos para 1000 iterações.

Conclusão e sugestões para trabalhos futuros

- Arquitetura: MLP, backpropagation;
- Ambiente de desenvolvimento: Python, pandas, pyBrain, Google Finance API, matplotlib, pyCharm IDE;
- Resultados significativos do modelo desenvolvido e treinado;
- Avaliação de outras topologias para o contexto do trabalho;
- Pesquisa avançada sobre fatores que influenciam nos resultados das ações;
- Métodos que atuem no reconhecimento de queda e altas das ações, onde a rede obteve erros significativos.